SYNTHESE

Détection d'intrusion dans un système informatique : méthodes et outils

Ludovic Mé — Véronique Alanou

SUPÉLEC
BP 28
35511 Cesson Sévigné Cedex

RÉSUMÉ. Nous appelons «intrusion» toute violation de la sécurité logique d'un système informatique. Le mécanisme d'audit de sécurité, consistant en l'enregistrement de tout ou partie des actions effectuées sur le système pour en faire une analyse a posteriori, permet de détecter des intrusions. Pour cela, deux approches sont actuellement utilisées : l'approche comportementale et l'approche par scénarios. Nous présentons ces deux approches et en donnons les avantages et inconvénients respectifs. Divers outils mettent en œuvre l'une ou l'autre de ces approches, voire les deux. Nous décrivons quatre de ces outils (NIDES, Hyperview, ASAX et Gassata) et donnons un tableau de synthèse indiquant la ou les approches utilisées par ces quatre outils et quelques autres. Dans la pratique, l'utilisation simultanée de plusieurs méthodes et outils semble indispensable afin de cumuler leurs avantages respectifs.

ABSTRACT. We call "intrusion" any violation of computer security in which software mechanisms are used rather than physical ones. The security audit mechanism, i.e., the recording and examination of system activities, allows intrusion detection. In this paper, the two current approaches for intrusion detection are discussed in detail: behavioral approach and rule based approach. We give some examples of tools implementing these approaches and conclude on the necessity to use several detection methods simultaneously in order to combine their individual advantages.

MOTS-CLÉS : Détection d'intrusion. Audit de sécurité.
KEY WORDS : Intrusion detection. Security audit.

1. Introduction

La sécurité d'un système informatique peut être abordée de plusieurs manières, qu'il est souvent indispensable de combiner pour atteindre un niveau de sécurité

suffisant :

1. l'approche préventive coercitive consiste à contraindre les usagers à respecter certaines procédures lors de toute utilisation du système. Le mécanisme le plus répandu dans ce cadre est la simple authentification par mot de passe. Dans le cas où un haut niveau de sécurité est demandé, on peut mettre en œuvre une politique de sécurité plus élaborée [ROL 94].
2. l'approche préventive analytique permet d'identifier les menaces pesant sur un système donné. On dispose d'outils qui, pour un type de système particulier, effectuent certains contrôles permettant de caractériser l'état de sécurité du système en question, par rapport à un état de référence défini précédemment. Ces outils sont dits «outils de contrôle statique». Les outils COPS (*Computer Oracle and Password System*) [FAR 91] et CAS (Conseiller Automatique en Sécurité) [DAC 91] font partie de cette catégorie. Une variante de ce type d'approche est proposée par les systèmes d'analyse à base de règles. Ces outils sont des systèmes experts auxquels l'utilisateur propose un but (par exemple s'approprier les droits en écriture sur le fichier `/etc/passwd` d'un système Unix, sachant qu'on leur confère initialement tels et tels droits) et ils tentent d'atteindre ce but par un enchaînement intelligent de commandes du système. Du succès ou de l'échec du défi, on tire des informations pertinentes sur la sécurité du système. L'outil KUANG (inclus dans COPS) entre dans cette famille.
3. l'approche curative consiste à enregistrer tout ou partie des actions effectuées sur le système pour en faire une analyse *a posteriori* permettant de démasquer les auteurs d'intrusions. Le mécanisme permettant l'enregistrement des actions est appelé «audit de sécurité».

Nous présentons dans le paragraphe 2 les activités liées à l'audit de sécurité. Dans le paragraphe 3, nous exposons les deux approches actuelles de la détection d'intrusion et en donnons les avantages et inconvénients respectifs. Dans le paragraphe 4, nous présentons quelques outils de détection d'intrusion et dressons un tableau comparatif indiquant les méthodes mises en œuvre par ces outils. Nous concluons sur la nécessité d'utiliser simultanément plusieurs méthodes de détection, de manière à cumuler leurs avantages respectifs.

2. Activités liées à l'audit de sécurité

Toute opération entreprise sur un système informatique se traduit par une séquence d'actions effectuées par le système. Ces actions sont appelées *activités système*. Une activité système intervenant à un certain moment est appelée *événement*. Un *journal d'audit de sécurité* est un fichier dans lequel est enregistré chronologiquement tout ou partie des événements. Les enregistrements du journal d'audit sont aussi appelés *traces d'audit*.

Le journal d'audit doit permettre, à partir de l'étude des séquences d'événements que l'on y trouve, de reconstituer les opérations entreprises par certains utilisateurs

spécifiques du système, dits utilisateurs audités. Il doit être possible de répondre aux six questions suivantes :

— quelle opération a été faite ?
— qui a fait l'opération ? Identifiant et label (niveau de sécurité et ensemble de domaine) du processus et de l'utilisateur pour le compte duquel il agit.
— quelles ressources du système ont-t-elles été affectées ? Lorsqu'il s'agit d'un fichier, on enregistre son nom et son chemin d'accès, ainsi que les données ajoutées ou supprimées.
— quand l'opération a-t-elle été réalisée (horodatage de l'enregistrement) ?
— où cela est-il arrivé ? Dans le cas où l'opération qui a conduit à l'enregistrement s'est produite sur un serveur distant, on enregistre l'identifiant de ce serveur.
— en cas d'échec, pourquoi l'opération a-t-elle échoué ?

Pour répondre à ces questions, l'officier de sécurité doit spécifier les utilisateurs et les activités système à auditer, assurer la collecte des événements dans le fichier d'audit et analyser régulièrement ce fichier. De plus il essayera de réparer les dégâts éventuellement détectés.

2.1. *Spécification des activités système à auditer*

Nous donnons ici une liste non exhaustive, inspirée de [BRO 79], des informations pertinentes à auditer. Selon le niveau de sécurité qu'il souhaite atteindre, l'officier de sécurité définira des événements auditables correspondant à certaines de ces informations.

2.1.1. *Informations sur les accès au système*

Il s'agit d'obtenir des informations sur ce qui constituera le point de départ de toute investigation sur une violation éventuelle de la sécurité :

— qui a accédé au système (identifiant de l'utilisateur ou du processus),
— quand (horodatage de l'accès au système),
— où (identifiant du terminal, adresse),
— comment (mode d'entrée : interactif, batch local, connexion distante).

2.1.2. *Informations sur l'usage fait du système*

Il s'agit de montrer quelles ressources du système ont été utilisées et comment :

— commandes systèmes utilisées et leur résultat (échec ou succès),
— accès à une unité d'entrée/sortie,
— utilisation de la CPU,
— taux d'occupation mémoire par utilisateur.

2.1.3. *Informations sur l'usage fait des fichiers*

C'est bien sûr un point très sensible puisque les fichiers contiennent l'information. Pour chaque accès à un fichier, on s'intéressera aux points suivants :

— horodatage de l'accès,
— type de l'accès (ouverture, fermeture, lecture, ajout d'enregistrement(s), modification d'enregistrement(s), purge du fichier, ...),
— source de l'accès (triplet (utilisateur, terminal, application)),
— volume d'informations échangées lors de l'accès.

2.1.4. *Informations relatives à chaque application*

Chaque application conduit à des événements qui peuvent influer sur la sécurité du système. On pourra enregistrer les événements suivants :

— les lancements et arrêts d'application,
— les modules réellement exécutés,
— les commandes exécutées et leurs résultats (échec ou succès),
— les données entrées,
— les sorties produites.

2.1.5. *Informations sur les violations éventuelles de la sécurité*

Il s'agit d'enregistrer tous les événements pour lesquels il y a eu une tentative d'accès à une ressource du système sans que cet accès soit autorisé par les règles de la politique de sécurité. Parmi ces événements, on peut citer :

— la tentative d'exécution d'une application dans un mode privilégié (par exemple la modification des droits d'accès sous Unix),
— la tentative d'accès à un fichier non autorisé ou la fourniture d'un mot de passe erroné pour cet accès,
— la tentative d'utilisation de certaines commandes du système réservées à des utilisateurs privilégiés,
— le changement des droits d'accès à des fichiers sensibles,
— l'accès au système à des moments ou depuis des lieux inhabituels. Il faut définir ce qui est habituel pour chaque utilisateur, un groupe d'utilisateurs, ou tous les utilisateurs. Les informations de ce type doivent être exploitées rapidement de manière à limiter les effets éventuels d'une atteinte à la politique de sécurité. Même si on a une exploitation en quasi-temps réel, on gardera une trace de ces informations pour en permettre une étude ultérieure.

2.1.6. *Informations statistiques sur le système*

Il est possible de repérer des activités anormales dans un système en observant attentivement quelques facteurs clé. Par exemple, on pourra tirer des conclusions des informations suivantes :

— niveau anormalement élevé des refus d'accès au système,

— niveau anormalement élevé ou bas de l'usage de certaines commandes du système (en particulier les commandes demandant des privilèges particuliers).

2.2. *Collecte des événements*

La plupart des systèmes d'exploitation disposent d'un sous-système d'audit capable de générer certains types d'événement. Le noyau du système assure alors la génération et la collecte de ces événements.

Il est également possible de générer des événements au sein des applications. Dans ce cas, on doit fournir au développeur une «boîte à outils sécurité» lui offrant les primitives de génération et de collecte adéquates. Le code source des applications devra être examiné pour vérifier que l'audit est bien réalisé.

Nous ne tenons pas à développer davantage l'aspect collecte dans cet article. Le lecteur intéressé se reportera au manuel d'utilisation du système d'exploitation sur lequel il souhaite faire de l'audit.

2.3. *Analyse du journal d'audit*

Le but de l'analyse est de révéler toute violation des règles de la politique de sécurité de manière à identifier les responsables, identifier les dégâts éventuels (et les réparer si possible) et proposer des changements dans les éléments qui assurent la sécurité du système. L'analyse doit permettre de détecter toutes sortes de transgressions, depuis l'intrusion dans le système par un utilisateur non autorisé jusqu'aux abus de toutes formes imputables aux utilisateurs connus du système. Cependant, l'audit de sécurité se fera parfois dans un ou plusieurs objectifs précis. Ainsi, on parle parfois d'audit des canaux cachés (détection de fuite par canal caché), d'audit des gains de privilèges ou d'audit des accès au système et des accès aux données (détection d'intrusion).

Nous nous intéressons ici à ce dernier cas. L'analyse du journal d'audit se fera donc dans l'objectif de découvrir des comportements des types suivants :

— Intrusions portant atteinte à la confidentialité :
  . vol de données : lecture, copie ou prise (copie suivie d'une destruction),
  . furetage : parcours des répertoires à la recherche d'informations dont on ne connait pas forcément l'existence,
  . fuite d'informations par canal caché,
  . inférence illégitime : corrélation illégitime entre des données pour lesquelles on a des droits d'accès.

— Intrusions portant atteinte à l'intégrité :
  . fraude : modification illégitime de fichiers de données,
  . introduction de programmes malveillants.

— Intrusions portant atteinte à la disponibilité de service :
  . destruction illégitime ou abusive de fichiers (données ou programmes),
  . occupation illégitime ou abusive de ressource(s) avec ou sans bénéfice d'usage,
  . réduction illicite ou abusive des droits des usagers.

De plus, les programmes malveillants (virus, cheval de Troie, ver, bombe logique) peuvent aussi bien porter atteinte à la confidentialité qu'à l'intégrité ou à la disponibilité de service.

Les actions qui peuvent être entreprises si on détecte une intrusion en cours, sont la déconnexion de l'intrus ou son confinement dans des répertoires particuliers, limitant les dommages possibles mais permettant d'étudier plus précisément son comportement.

2.3.1. *Fréquence de l'analyse des traces d'audits*

Les analyses doivent être fréquentes afin que le minimum de malversations reste indétecté. Les analyses journalières, préconisées il y a quelques années [BRO 79], semblent aujourd'hui insuffisantes. L'objectif à atteindre est la surveillance du système en quasi-temps réel. A ce titre, on peut envisager que l'écart entre deux analyses successives des traces d'audit soit réduit à quelques minutes, voire à quelques dizaines de secondes (ce doit être un élément de la politique de sécurité). Cette contrainte a des conséquences fortes sur les méthodes d'analyse utilisables, qui doivent permettre de traiter très rapidement des volumes de données considérables.

2.3.2. *Protection du journal d'audit*

La confidentialité, l'intégrité et la disponibilité du journal d'audit sont essentielles, un utilisateur malicieux ne devant pas pouvoir effacer ou modifier les traces de ses opérations. Le fichier d'audit doit donc être protégé contre toute lecture ou modification par des utilisateurs autres que ceux qui sont autorisés.

2.3.3. *Le cas des réseaux*

Lorsque l'on souhaite faire de l'audit sur un système distribué, il est impératif de disposer d'une base de temps commune qui permettra d'estampiller les événements survenants sur toutes les machines connectées. Il faut donc construire un temps global sur le réseau [ECM 88] ou accepter l'utilisation d'une fenêtre de temps [KIM 91]. Par ailleurs, il faut être capable de diffuser les types d'événement à auditer et de collecter les enregistrements d'audit en provenance des différentes machines connectées sur le réseau, de manière à constituer un journal d'audit global. Ces transferts d'information doivent être sécurisés, par exemple par un mécanisme de chiffrement.

3. Méthodes de détection d'intrusion

Dans chacune des phases de l'audit, l'expérience et la compétence de l'officier de sécurité sont essentielles.

Pour mener à bien la phase de spécification des événements et des utilisateurs à auditer, il doit avoir tout à la fois une vision globale et une vision précise du système,

de manière à définir correctement, c'est-à-dire sans effet de «passoire», les activités système à auditer.

Dans la phase d'analyse du journal, le travail de l'officier de sécurité est énorme. En effet, les systèmes d'exploitation actuels génèrent un tel volume d'informations (même lorsque le nombre d'activités auditées est faible) qu'il est quasiment impossible de traiter manuellement les informations du fichier d'audit. Dans la pratique, les traces d'audit ne sont utilisées qu'en cas de problèmes graves, pour tenter d'en déterminer la cause. En d'autres termes, les traces d'audit ne sont pas actuellement utilisées pour faire de la détection d'intrusion, sauf dans des environnements très particuliers, hautement sécurisés.

Il est donc particulièrement nécessaire d'offrir à l'officier de sécurité un outil ou une panoplie d'outils d'aide à l'analyse du journal d'audit. Nous présentons dans ce paragraphe les méthodes sur lesquelles les prototypes de détection d'intrusion existants s'appuient. Ces méthodes sont de deux types car un attaquant qui cherche à s'introduire dans un système peut exploiter des vulnérabilités qui peuvent être connues ou inconnues de l'officier de sécurité du système.

3.1. *Détection d'une attaque exploitant une vulnérabilité inconnue : approche comportementale*

Une approche, proposée par J.P. ANDERSON [AND 80] puis reprise et étendue par D.E. DENNING [DEN 87], consiste à utiliser des méthodes basées sur l'hypothèse selon laquelle l'exploitation d'une vulnérabilité du système implique un usage anormal de celui-ci. Une intrusion est donc identifiable en tant que déviation par rapport au comportement habituel d'un utilisateur. Voici quelques exemples étayant cette hypothèse :

- un essai d'intrusion par un utilisateur non connu du système donnera lieu à un taux anormal de mots de passe erronés,
- l'attaquant par déguisement se connecte à une heure inhabituelle, il utilise abondamment des commandes lui permettant de changer de répertoire, peut-être n'utilise-t-il jamais l'utilitaire favori de l'utilisateur habituel,
- un utilisateur connecté légitimement au système et qui essaye de contourner la politique de sécurité se connectera la nuit, exécutera des programmes qu'il n'a pas l'habitude d'utiliser, donnera lieu à un plus grand volume d'enregistrements d'audit, utilisera une imprimante sur laquelle il ne sort généralement pas de document, etc.
- un cheval de Troie différera du programme légal dont il a pris la place en terme d'utilisation des ressources d'entrée/sortie,
- une attaque par déni de service donnera lieu à un taux d'utilisation anormalement élevé de certaines ressources du système.

Bien sûr, les phénomènes décrits dans ces exemples peuvent avoir une autre cause qu'une attaque du système, par exemple un changement de fonction de l'utilisateur au sein de l'entreprise. On s'attachera donc à trouver des méthodes possédant le plus fort taux de discrimination possible (c'est-à-dire ayant le plus fort taux de détection

d'intrusion et le plus faible taux de fausses alarmes). De plus, on se réfèrera à un seuil au-delà duquel on considérera que le comportement est intrusif.

Cette approche, dont la question de base est «le comportement actuel de l'utilisateur est-il cohérent avec son comportement passé ?», est appelée approche comportementale. Pour caractériser le comportement normal d'un utilisateur (on parle de modèle de comportement), l'approche la plus immédiate consiste à utiliser des méthodes statistiques [1]. Il est également possible d'envisager l'utilisation de systèmes experts ou de réseaux de neurones. Nous présentons successivement ces trois approches dans la suite de ce paragraphe.

3.1.1. *Méthodes statistiques : le modèle de Denning*

Le modèle de Denning est un modèle de comportement qui se compose de six éléments :

— **les sujets :** ce sont les initiateurs de toute activité observée sur le système, c'est-à-dire les utilisateurs ou les processus agissant pour leurs comptes.
— **les objets :** ce sont toutes les ressources du système (fichiers de données, fichiers de programme, messages, périphériques, ...). La granularité des objets sera plus ou moins fine selon le degré de sécurité recherché : dans certains cas chaque fichier donnera lieu à la définition d'un objet, dans un autre il faudra descendre au niveau des enregistrements, dans un troisième cas le niveau répertoire sera suffisant.
— **les enregistrements d'audit :** ils sont générés par le système lors de toute action entreprise par un sujet sur un objet. Ils se composent des informations suivantes :
. le sujet ayant entrepris l'action,
. le type de l'action (par exemple login, lecture, écriture, ...),
. les objets affectés (une action peut impliquer plusieurs objets),
. échec ou succès de la commande,
. des éléments quantitatifs sur l'action (par exemple nombre d'octets ou d'enregistrements transférés dans une copie de fichier),
. un horodatage de l'action.
— **les profils :** ce sont des structures qui caractérisent le comportement des sujets vis-à-vis des objets, par l'intermédiaire de valeurs statistiques résultant de l'observation des sujets. Un profil caractérise le comportement d'un utilisateur ou d'un groupe d'utilisateurs envers un objet ou un ensemble d'objets. Il donne une vue synthétique des actions des utilisateurs sur les objets.
— **les enregistrements d'anomalie :** ils sont générés quand une activité anormale est détectée,
— **les règles d'activité :** elles définissent les actions à entreprendre lorsque certaines conditions sont remplies sur les enregistrements d'audit ou les enregistrements

1. Il peut sembler intéressant de décrire statistiquement les comportements possibles d'un attaquant, de manière à identifier par la suite tout comportement intrusif par comparaisons avec ces «profils d'attaquant». Malheureusement, on dispose de trop peu de données sur le sujet pour que la construction de tels profils soit envisageable.

d'anomalie. Elles ont la forme classique conditions-actions. Les actions peuvent par exemple consister à alerter l'officier de sécurité.

Un profil est constitué par un ensemble de variables représentant une quantité accumulée pendant une certaine période de temps (minute, heure, journée, semaine, ... ou intervalle entre deux événements audités particuliers, par exemple entre connexion et déconnexion). Voici quelques exemples de variables possibles :

— nombre de mauvais mots de passe saisis en une minute,
— nombre, en millièmes de seconde, de quantums de temps CPU occupés par un programme entre son lancement et sa terminaison,
— nombre de fois qu'une commande système particulière est exécutée par un utilisateur donné, pendant le temps ou il est connecté,
— etc.

Le modèle statistique permet de déterminer, au vu de n observations x_1, ..., x_n faites sur une variable x, si la valeur x_{n+1} de la (n+1)^{ème} observation est normale ou non. DENNING [DEN 87] propose plusieurs modèles :

— **Comparaison de la nouvelle valeur de x avec une limite fixe :** dans ce cas les n observations précédentes ne sont utiles que pour se donner une idée de la limite en question.
Exemple : on considère qu'il y a essai d'intrusion si on mesure dix mots de passe erronés en une minute.
— **Utilisation de la moyenne et de l'écart type des n observations précédentes :** l'observation $n+1$ est considérée comme anormale si x_{n+1} sort de l'intervalle de confiance défini par un écart de $(\pm D \times \sigma)$ autour de la moyenne.
Ce modèle présente l'avantage d'être capable d'apprendre ce qui est «normal» à partir des observations passées. Pour cela, on met à jour la moyenne et l'écart type à chaque nouvelle observation en pondérant les observations de manière à ce que les plus récentes aient le plus fort poids.
— **Utilisation des covariances :** ce modèle est similaire au précédent mais il permet de combiner plusieurs variables afin d'en tirer une synthèse. Il permet d'exploiter le fait que deux mesures (ou plus) représentent des manières différentes de caractériser le même aspect du comportement d'un utilisateur.
— **Utilisation des processus de Markov :** ce modèle permet de définir la probabilité du passage d'un état, défini par le contenu d'un enregistrement d'audit, à un autre état, également défini par un enregistrement d'audit. Est considéré comme anormal un enregistrement qui apparaît et dont la probabilité, au regard des états précédents, est trop faible. Cette approche est intéressante pour les attaques dans lesquelles est requis un enchaînement de commandes dans un certain ordre.
— **Utilisation des séries temporelles :** une nouvelle observation est anormale si sa probabilité d'apparition, au moment où elle apparaît, est trop faible.

DENNING envisage la possibilité d'utiliser d'autres modèles, utilisant plus que les deux premiers moments mais moins que l'ensemble complet des enregistrements d'audit.

3.1.2. *Systèmes experts*

Pour représenter l'usage «normal» qu'un utilisateur fait du système, il est possible d'utiliser un ensemble de règles au lieu d'un modèle statistique. Cela permet d'utiliser un système expert comme outil de détection d'intrusion.

Les règles d'un tel système expert peuvent être, soit entrées manuellement, soit générées automatiquement à partir des enregistrements d'audit. L'entrée manuelle sera par exemple utilisée pour exprimer une politique de sécurité. Les règles générées décrivent quant à elle des comportements.

Les systèmes experts présentent un inconvénient souvent cité : la base de règles est ni simple à créer, ni simple à maintenir.

3.1.3. *Réseaux de neurones*

On peut envisager l'application des réseaux de neurones à la détection d'intrusion de plusieurs manières :

— Pour donner une modélisation statistique du comportement des utilisateurs [LUN 90a]. On est alors très proche des méthodes statistiques telles que celles présentées ci-dessus, l'avantage des réseaux de neurones étant qu'il n'est pas nécessaire de faire d'hypothèses sur les variables aléatoires.
— Pour classifier le comportement des utilisateurs [LUN 90a, FOX 90] par un algorithme de type «carte de Kohonen» (classification automatique et non supervisée).
— Pour prédire le comportement des utilisateurs [DEB 92, DEB 93]. Le réseau apprend les séquences de commandes usuelles à chaque utilisateur. Il lui est alors possible, après chaque commande passée par l'utilisateur, de prédire la commande suivante sur la base de ce qu'il a appris. En cas de déviation entre la prévision et la réalité, une alarme est émise.

Il faut cependant noter :

— qu'un réseau de neurones ne fournit pas d'explication sur le raisonnement l'ayant amené à proposer un diagnostic d'intrusion,
— que le paramétrage d'un réseau de neurones est délicat et peut influer considérablement sur les résultats fournis.

3.2. *Détection d'une attaque exploitant une vulnérabilité connue : approche par scénarios*

S'il n'est pas envisageable de décrire statistiquement le comportement d'un attaquant, il est possible de donner des règles sur sa manière de procéder. Ces règles prennent la forme de scénarios d'attaque exploitant des vulnérabilités du système précédemment identifiées. On parle parfois pour cette approche, dont la question de base est «le comportement de l'utilisateur correspond-il à une attaque connue ?», de modèle de scénario. Avec de tels systèmes, toute attaque non prévue reste indétectée.

3.2.1. *Systèmes experts*

Dans l'optique d'une détection d'attaque exploitant une vulnérabilité connue, il est possible d'utiliser un système expert dont la base de connaissances contient :

— des règles précisant ce qui est suspect *a priori* (en fonction de la politique de sécurité en vigueur sur le site surveillé),
— des règles concernant les failles de sécurité connues (suite à de précédentes intrusions ou suite à un avis du CERT (*Computer Emergency Response Team*) par exemple),
— des règles codant le savoir-faire d'un officier de sécurité en matière de détection d'intrusion.

3.2.2. *Pattern Matching (reconnaissance de formes)*

Chaque événement d'un scénario d'attaque peut être considéré comme une lettre prise dans un alphabet constitué par l'ensemble des événements auditables sur le système cible. Le fichier d'audit peut être vu comme une chaîne de caractères principale et les scénarios d'attaque comme des sous-suites qu'il s'agit de localiser dans cette chaîne principale. Le problème de détection d'intrusion se ramène alors à un problème de *pattern matching*.

De cette analogie, découle la nécessité de disposer d'un langage de description des scénarios d'attaque et d'un algorithme de *pattern matching* efficace.

3.2.3. *Algorithmes génétiques*

Les algorithmes génétiques (G.A.), proposés par JOHN HOLLAND dans les années 70 [HOL 75], s'inspirent de l'évolution génétique des espèces, plus précisément du principe de sélection naturelle. Il s'agit d'algorithmes de recherche d'optimum construits en s'inspirant de la nature dans laquelle vivent, dans des conditions parfois difficiles, des organismes robustes et adaptables.

La fonction dont on recherche l'optimum est dite fonction objective. On ne fait aucune hypothèse sur cette fonction, en particulier elle n'a pas à être dérivable, ce qui représente un avantage sur certaines méthodes de recherche d'extremum (par exemple les méthodes basées sur le gradient). Pour résoudre un problème quelconque par approche génétique, on devra l'exprimer sous la forme d'une fonction objective.

Un algorithme génétique manipule une population de taille constante, formée d'individus représentant chacun le codage d'une solution potentielle au problème à résoudre. Chaque individu prend la forme d'une chaîne de caractères. Dans les cas classiques, l'alphabet chromosomique est binaire (les deux allèles possibles sont 0 et 1) mais parfois les spécificités du problème à résoudre amènent à choisir un alphabet de cardinalité supérieure à 2.

La population évolue en générations successives. La taille constante de la population induit un phénomène de compétition entre les individus, les plus forts survivant et se reproduisant entre eux pour créer de nouveaux individus, les plus faibles dispa-

raissant petit à petit. De plus, lors des créations d'individus, des mutations génétiques (c'est-à-dire la modification d'un caractère dans la chaîne) se produisent.

Cette analogie avec la nature conduit à définir les trois opérateurs génétiques de base : sélection, recombinaison et mutation.

La traduction algorithmique des adjectifs «faible» et «fort» appliqués aux individus conduit à définir une fonction sélective qui permet d'associer une valeur dite sélective à chaque individu de la population. La fonction sélective f est souvent une transformation o de la fonction objective ($f(x) = g(o(x))$).

L'application des opérateurs génétiques sur des individus jugés par une fonction sélective particulière permet d'explorer l'espace des solutions à la recherche d'un extremum. Cette recherche se fait avec un très bon équilibre entre l'exploitation des résultats déjà acquis et l'exploration de nouvelles régions de l'espace, ce qui permet d'obtenir un très faible rapport entre le nombre de points échantillonés et le nombre de points total dans l'espace de recherche.

Mé [Me94] donne un exemple d'application des algorithmes génétiques à la recherche de scénarios d'attaque prédéfinis dans les traces d'audit, chaque scénario étant défini comme un ensemble d'événements. Tous les entremêlements possibles de ces scénarios doivent être pris en compte. On a là un phénomène d'explosion combinatoire qui interdit l'utilisation d'algorithmes de recherche classiques. L'approche génétique, grâce à son bon équilibre exploration/exploitation, permet d'obtenir en un temps de traitement raisonnable (quelques secondes à quelques minutes selon le nombre d'attaques considérées sur un IBM RS6000 320) le sous-ensemble des attaques potentiellement présentes dans les traces d'audit.

3.3. *Approche comportementale ou approche par scénarios ?*

Chacune de ces deux approches présente des avantages et des inconvénients.

3.3.1. *Avantage de l'approche comportementale*

— la détection d'intrusion inconnue est possible.

3.3.2. *Avantage de l'approche par scénarios*

— la prise en compte des comportements exacts des attaquants potentiels est possible.

3.3.3. *Inconvénients de l'approche comportementale*

— le choix des différents paramètres du modèle statistique est assez délicat et soumis à l'expérience de l'officier de sécurité,
— l'hypothèse d'une distribution normale des différentes mesures n'est pas prouvée,
— le choix des mesures à retenir pour un système cible donné est délicat,
— en cas de profonde modification de l'environnement du système cible, le modèle statistique déclenche un flot ininterrompu d'alarmes, du moins pendant une période transitoire,

— un utilisateur peut changer lentement de comportement dans le but d'habituer le système à un comportement intrusif,
— il est difficile de dire si les observations faites pour un utilisateur particulier correspondent à des activités que l'on voudrait prohiber,
— pour un utilisateur au comportement erratique, toute activité est «normale». Une attaque par déguisement sur son compte ne pourra pas être détectée,
— il n'y a pas de prise en compte des tentatives de collusion entre utilisateurs, alors même que cet aspect est très important, notamment dans le cas des réseaux.

3.3.4. *Inconvénients de l'approche par scénarios*

— la base de règles doit être bien construite, ce qui est parfois délicat,
— les performances du système expert sont limitées par celles de l'expert humain qui a fourni les règles. Or les connaissances des officiers de sécurité en matière de détection d'intrusion sont relativement faibles car, la plupart du temps, le volume énorme des fichiers d'audit les a découragés de toute analyse.

Il semble donc indispensable d'utiliser simultanément une approche comportementale et une approche par scénarios de manière à profiter des avantages de l'une et de l'autre. C'est ce qui a été fait dans l'outil NIDES que nous présentons dans cet article.

4. Quelques outils de détection d'intrusion

Nous présentons dans ce paragraphe les outils NIDES, Hyperview, ASAX et GASSATA. NIDES, développé par SRI (*Stanford Research Institute*) est l'outil de référence. Il intègre un module statistique et un système expert. Hyperview a été développé par la société française CS Telecom. Au module statistique et au système expert, Hyperview ajoute un module neuronal. ASAX, développé conjointement par l'Université de Namur et Siemens Nixdorf S.A., offre un langage de description de scénarios d'attaque et un moteur de type système expert pour l'analyse. GASSATA est, quant à lui, un prototype développé à Supélec. Il s'agit d'un algorithme génétique qui recherche des scénarios prédéfinis dans les traces d'audit.

4.1. *NIDES*

Développé sur financement de l'*U.S. Navy*, NIDES[2] (*Next generation Intrusion Detection Expert System*) est un outil de détection d'intrusion qui s'appuie sur les deux approches présentées ci-dessus : une approche statistique (modèle de DENNING) pour les attaques exploitant les vulnérabilités inconnues et une approche système expert pour celles exploitant les vulnérabilités connues [JAV 93, JAV 94, AND 94].

NIDES fonctionne sur une machine dédiée, indépendante du système surveillé, auquel elle est reliée par un réseau. Le système surveillé, appelé système cible, produit

2. NIDES est le nom donné à la cinquième génération de cet outil. Les quatre premières générations portait le nom IDES [LUN 88, LUN 90b, LUN 90c, GAR 91].

des enregistrements d'audit, qu'il transmet à NIDES par le réseau, après les avoir convertis au format NIDES et chiffrés. L'avantage que présente cette approche est d'avoir un impact acceptable, bien qu'il reste à évaluer précisément, sur les performances du système cible. Le système cible est un réseau de stations Sun. La machine dédiée est également une station Sun et elle communique avec chaque station du système cible en utilisant SunRPC sur un canal TCP/IP. Les principes de NIDES restent cependant totalement indépendants du système cible (SRI a développé une version pour mainframe).

NIDES exploite les enregistrements du fichier d'audit qui lui parviennent dans le but d'apprendre les habitudes des utilisateurs ou groupes d'utilisateurs (et de s'adapter à toute modification de ces habitudes) de manière à détecter (et à rapporter à l'officier de sécurité) les activités anormales des utilisateurs.

Les événements compris par NIDES sont essentiellement les suivants :

— accès aux fichiers :
  . création ou destruction de fichier,
  . ouverture d'un fichier (en écriture ou en lecture),
  . écriture ou lecture dans un fichier,
  . renommage d'un fichier,
  . modification des modes d'accès à un fichier,
  . modification du propriétaire d'un fichier,
— accès aux répertoires :
  . création ou suppression de répertoire,
  . changement de répertoire courant,
— connexions/déconnexions :
  . connexion ou déconnexion d'un utilisateur,
  . échec d'un essai de connexion,
  . passage en super-utilisateur,
  . échec d'un essai de passage en super-utilisateur,
— consommation de ressource :
  . CPU, mémoire, E/S,
— exécution d'une commande du système,
— activité réseau,
  . telnet, rlogin, ftp, etc.
— activités des programmes.

4.1.1. *L'approche statistique de NIDES*

Elle s'appuie sur deux types de mesures permettant de caractériser l'activité des utilisateurs [LUN 90c] :

— les mesures continues contiennent les aspects quantifiables du comportement des utilisateurs (par exemple nombre d'enregistrements d'audit produits en une heure, nombre de fichiers accédés, temps CPU utilisé par jour, ...),

— les mesures catégorielles nécessitent pour leur part que soit précisé un nom de ressource utilisée. Ainsi, dans la catégorie des accès aux fichiers, on devra préciser les noms des fichiers accédés (par exemple nombre d'accès à un fichier donné, nombre d'utilisation des différents éditeurs de texte disponibles sur le système, ...).

Supposons que n mesures $X_1, X_2, ..., X_n$ soient pertinentes pour évaluer le comportement des utilisateurs du système cible. A chaque variable X_i est associée une densité de probabilité Q_i. L'intervalle des valeurs possibles de X_i est divisé en seize sous-intervalles croissants géométriquement avec un facteur 2 (Q_i est donc constituée de seize fréquences) (ce découpage peut être affiné lorsqu'on acquiert plus d'expérience sur le système cible). Les densités de probabilité sont mises à jour quotidiennement selon un algorithme qui permet de tenir compte du passé en favorisant ce qui est proche.

On se reportera à [LUN 90c] ou à [Me92] pour plus de détails.

Les diverses variables aléatoires X_i peuvent être exprimées dans des unités quelconques (quantum de temps, nombre d'accès fichier par minute, nombre de pages imprimées par jour, ...). De manière à pouvoir synthétiser l'ensemble des informations apportées par les diverses variables en un résumé statistique (RS) qui donne le degré de normalité du comportement de l'utilisateur, on associe à chaque variable X_i une grandeur D_i qui s'exprime dans la même unité pour toutes les variables X_i. D_i est proche de zéro lorsque la valeur observée de X_i est normale (c'est-à-dire conforme au passé) et D_i croît lorsque la valeur observée de X_i s'éloigne de ce qui est normal. De cette manière, plus l'enregistrement d'audit est proche du comportement antérieur de l'utilisateur, plus son résumé statistique est faible (il est même nul dans le cas où tous les $X_i$sont dans l'intervalle pour lequel leur probabilité est la plus forte). Il est alors possible de définir des seuils tels que par exemple :

— si $RS <$ 22, pas de problème,
— si 22 $\leq RS \leq$ 28, alerte jaune,
— si $RS >$ 28, alerte rouge.

RS est calculé pour chaque enregistrement d'audit. Cela signifie que si un utilisateur génère N_{ea} enregistrements par jour, l'officier de sécurité dispose de N_{ea} résumés statistiques qualifiant le comportement de cet utilisateur. Puisqu'ils dépendent du passé, ces N_{ea} résumés ne sont pas indépendants. Ils évoluent lentement dans un sens ou dans l'autre. De manière à ne pas submerger l'officier de sécurité, NIDES émet une alerte, soit lorsque le résumé d'un utilisateur change de zone, soit lorsque le résumé est en zone jaune ou rouge depuis un certain temps.

4.1.2. *Le système expert de NIDES*

NIDES intègre un système expert qui évalue le comportement des utilisateurs d'après une base de règles décrivant des comportements anormaux.

Le système expert de NIDES évalue les enregistrements d'audit au moment où ils sont produits. Il vérifie si certains champs de l'enregistrement correspondent à ce qui est spécifié dans les règles. Si c'est le cas, la règle correspondante est déclenchée.

Chaque règle déclenchée accroît un «taux de suspicion» (qui est propre à chaque utilisateur). Lorsque le taux dépasse un certain seuil, un rapport d'anomalie est émis vers l'officier de sécurité.

NIDES est logiciel gratuit fonctionnant sur un système Unix (SunOS). Cependant, la personnalisation du système expert pour un site donné nécessite une négociation commerciale. Une version pour mainframe a également été réalisée sur demande par le SRI. NIDES est actuellement utilisé par le FBI (*Federal Bureau of Investigations*), l'*U.S. Navy*, etc.

4.2. *Hyperview*

L'outil Hyperview a été développé par la société CS Telecom à partir de travaux menés depuis le début des années 90 [GAU 91, DEB 92, DEB 93]. Hyperview est utilisé dans un réseau de machines Unix. Il centralise les données fournies par chaque système surveillé, fait de la détection d'intrusion en quasi-temps réel et fournit des alarmes à l'officier de sécurité.

Hyperview est constitué de quatre modules : un module d'analyse statique, un module statistique, un module neuronal et un système expert.

4.2.1. *Le module d'analyse statique*

Hyperview permet d'automatiser la vérification de points sensibles ou les failles potentielles dans la configuration et l'administration des systèmes sous surveillance. Un module, appelé *Distributed Diagnostic Tool* (DDT), effectue des contrôles portant par exemple sur l'intégrité du système d'exploitation, les droits d'accès aux fichiers, les mots de passe ou la configuration réseau.

Les fonctionnalités de DDT sont à rapprocher de celles de COPS. Toutefois, certaines fonctions limitées de COPS ont été améliorées en s'inspirant d'outils tels que Crack [MUF 92] (contrôle des mots de passe) et Tiger [SAF 93].

4.2.2. *Le module statistique*

Ce module permet de modéliser le comportement des utilisateurs grâce aux données issues des mécanismes de comptabilité et d'audit. Il est possible de détecter les déviations de comportement des utilisateurs par rapport aux profils statistiques construits.

Le module statistique d'Hyperview a des objectifs plus modestes que celui de NIDES, la fonction de modélisation fine du comportement étant laissée au module neuronal. Le modèle statistique est un simple modèle à seuil prenant en compte les dépassements de moyennes statistiques précédemment établies. Il a pour but de détecter les vols de ressource, c'est-à-dire les utilisations «abusives» ou en tous cas inhabituelles des ressources du système (CPU, entrées-sorties, ...).

4.2.3. *Le module neuronal*

Le réseau de neurones modélise le comportement des utilisateurs par les séquences de commandes que ceux-ci utilisent. Il se distingue donc du module statistique présenté ci-dessus qui ne prend en compte que les utilisations isolées des commandes. En phase d'apprentissage, le réseau apprend, pour chaque utilisateur, les séquences qui lui sont habituelles. En phase d'exploitation, il lui est donc possible de prédire la commande que l'utilisateur devrait saisir au vu des séquences apprises. Si la commande saisie diffère de la prédiction, le comportement est jugé anormal.

Le réseau de neurones implémenté par Hyperview est un réseau récurrent simple avec architecture de GENT [GEN 92]. Ce type de réseau a été choisi car il est bien adapté à la prédiction de séries temporelles.

4.2.4. *Le système expert*

La base de règles du système expert d'Hyperview porte sur l'enchaînement des appels systèmes et sur la référence à des objets sensibles. Ce système fournit des alarmes de type «acquisition de privilèges» ou «changement de droits d'accès».

Le système expert traite directement les appels système enregistrés dans les fichiers d'audit de sécurité. Le système expert utilisé est IlogRules. Sur SunOS 4.1.3, environ 60 règles sont nécessaires pour réaliser la surveillance de 120 ressources sensibles.

L'ensemble des modules décrits ci-dessus permet de fournir à l'officier de sécurité des niveaux de risques par utilisateur et des niveaux de risques par système surveillé. Hyperview est implanté sur station Sun sous SunOS 4.1.3. La gestion de toutes les données est réalisée grâce au SGBD relationnel Ingres.

Hyperview est un logiciel commercial fonctionnant actuellement dans plusieurs contextes comme celui de la surveillance d'un serveur d'application (pour un grand compte du domaine télécom) ou celui de la surveillance de systèmes Unix (pour un grand compte du domaine spatial).

4.3. ***ASAX***

ASAX (*Advanced Security audit trail Analysis on uniX*), développé par l'Université de Namur et Siemens Nixdorf S.A., offre un langage à base de règles (RUSSEL) qui permet d'associer à un scénario d'attaque, une action à entreprendre si ce scénario se réalise [HAB 92].

ASAX définit un format standard de données d'audit (NADF, *Normalized Audit Trail Format*) vers lequel sont traduites les traces d'audit au format système. Les traces standard ainsi générées sont analysées séquentiellement en une seule passe car à un instant donné, un ensemble de règles est actif et représente l'ensemble des connaissances précedemment acquises.

ASAX (grâce à RUSSEL) permet donc de reconnaître des séquences particulières dans les traces d'audit et de déclencher les actions en conséquence.

ASAX est un prototype distribué gratuitement.

4.4. *GASSATA*

GASSATA est un prototype de détection d'intrusion construit autour d'un algorithme génétique implémentant les trois opérateurs de base définis par John HOLLAND : sélection proportionnelle, crossover un-point et mutation simple. Il s'appuie sur les résultats de [Me94] et a été développé sur une machine IBM RS6000 sous AIX 3.2.5 / X11R5.

Contrairement à ASAX dans lequel une attaque est définie par une séquence d'événements, GASSATA définie une attaque comme un ensemble de couples (événement d'audit, nombre d'occurrences de cet événement durant la session d'audit). Ces attaques sont regroupées dans une matrice appelée «matrice attaques/événements». L'algorithme génétique détermine le sous-ensemble d'attaques potentiellement présentes dans le fichier d'audit (si N attaques sont définies, on peut construire 2^N sous-ensembles). Chaque individu de la population que manipule l'algorithme génétique correspond à un sous-ensemble d'attaques particulier. La valeur sélective d'un individu est proportionnelle à son degré de réalisme au vu du fichier d'audit. L'algorithme génétique ne faisant survivre que les individus de forte valeur sélective, la polulation finale est constituée de clones de l'individu codant l'hypothèse la plus réaliste.

L'ordre temporel des événements dans le fichier d'audit n'est pas pris en compte. Il en résulte une altération de la pertinence de la détection. Cependant, on peut y voir un avantage car, dans la pratique, l'ordonnancement total des événements est bien souvent difficile, voire impossible, à réaliser. C'est particulièrement vrai lorsque l'unité de temps minimale offerte par le système d'audit est trop importante. Si l'audit s'étend à tout un réseau, le problème s'accentue puisqu'il faut alors construire un temps global, dont on sait la précision limitée.

GASSATA est un prototype en cours de finalisation.

4.5. Synthèse sur les outils

Un certain nombre d'outils reprennent les principes de IDES. On pourra se référer à [MCA 90, POR 92, DEB 93] pour une présentation rapide des outils suivants :

— AudES (*Auditing Expert System*) [TSU 88, TSU 90],
— *Computer Watch Audit Trail Analysis Tool* de AT&T,
— DIDS (*Distributed Intrusion Detection System*) de Lawrence Livermore National Laboratory, de Haystack Laboratories et de l'Université de Californie-Davis pour l'U.S. Air Force [STA 91],
— HAYSTACK (Haystack Laboratories pour le compte du Los Alamos National Laboratory) [SMA 88],
— ISOA (*Information Security Officier's Assistant*) [WIN 90, WIN 92],
— MIDAS (*Multix Intrusion Detection and Alerting System*) [SEB 88],
— NADIR (*Network Anomaly Detection and Intrusion Reporter*) du Los Alamos National Laboratory [JAC 91],
— NIDX (*Network real-time Intrusion Detection Expert system*) [BAU 88],

— W&S (*Wisdom and Sense*) du Los Alamos National Laboratory [VAC 89].

Par ailleurs, certains outils utilisent d’autres approches, comme IDIOT (*Intrusion Detection In Our Time*) [KUM 95] qui est basé sur le *pattern matching*.

Nous avons regroupé dans la table 1, un certain nombre d’outils de détection d’intrusion, en indiquant les méthodes utilisées par chacun à notre connaissance.

Outil	Méthode statistique	Système expert	Réseau de neurones	Pattern Matching	Algorithme génétique
NIDES	X	X			
AudES		X			
Computer Watch		X			
DIDS	X	X			
HAYSTACK	X	X			
ISOA	X	X			
MIDAS	X				
NADIR	X	X			
NIDX	X	X			
W&S	X				
Hyperview	X	X	X		
ASAX		X		X	
IDIOT				X	
GASSATA					X

Table 1. *Outils de la détection d’intrusion*

5. Conclusion

Dans cet article, nous avons présenté un état de l’art en matière de détection d’intrusion à partir des enregistrements d’audit de sécurité.

Le volume de données généré par les mécanismes d’audit des sytèmes actuels est très important (de l’ordre du méga-octet par utilisateur et par heure sur les systèmes Unix comme SunOS et AIX d’IBM). Il est donc indispensable d’offrir aux officiers de sécurité des méthodes et des outils leur permettant d’en extraire les informations utiles.

Deux grandes approches existent pour cela : l’approche comportementale et l’approche à base de règles. Chacune d’entre elles présentant des avantages et des inconvénients, leur utilisation simultanée est nécessaire.

Les outils existant mettent en œuvre l’une ou l’autre de ces méthodes, voire les deux. Il est du ressort de l’officier de sécurité de choisir le ou les outils les plus appropriés non seulement au système informatique, mais aussi à la politique de sécurité et au

niveau d'expertise des intrus potentiels. L'expérience et la compétence de l'officier de sécurité sont ici essentielles.

Pour terminer, il faut noter qu'en amont de l'analyse, la collecte des événements d'audit n'est pas sans perturber le travail des utilisateurs. En effet, les mécanismes de collecte des systèmes Unix actuels sont particulièrement gourmands en temps CPU et en accès disque.

6. Bibliographie

[AND 80] J.P. ANDERSON. « Computer Security Threat Monitoring and Surveillance ». Rapport Technique, James P. Anderson Company, Fort Washington, Pennsylvania, April 1980.

[AND 94] Debra ANDERSON, Teresa LUNT, Harold JAVITZ, Ann TAMARU, et Alfonso VALDES. « Safeguard Final Report: Detecting Unusual program Behavior Using the NIDES Statistical Component ». Rapport Technique, SRI, 1994.

[BAU 88] D.S. BAUER et M.E. KOBLENTZ. « NIDX, a Real-Time Intrusion Detection Expert System ». Dans *Proceedings of the USENIX'88 Conference*, pages 261–272, June 1988.

[BRO 79] P. BROWNE. « The Security Audit ». Dans *Checklist For Computer Security Center Self-Audits*, pages 173–184, 1979.

[DAC 91] Marc DACIER et Michel RUTSAERT. « Comment gérer la transitivité en sécurité ». Dans *Actes de la convention UNIX'91*, pages 205–217, 1991.

[DEB 92] Hervé DEBAR, Monique BECKER, et Didier SIBONI. « A Neural Network Component for an Intrusion Detection System ». Dans *Proceedings of the IEEE Symposium of Research in Computer Security and Privacy*, pages 240–250, May 1992.

[DEB 93] Hervé DEBAR. « *Application des réseaux de neurones à la détection d'intrusions sur les systèmes informatiques* ». Thèse de doctorat, Université de Paris 6, 1993.

[DEN 87] D.E. DENNING. « An Intrusion-Detection Model ». *IEEE transaction on Software Engineering*, 13(2):222–232, 1987.

[ECM 88] ECMA. « Security In Open Systems: A Security Framework », 1988.

[FAR 91] Dan FARMER et E.H. SPAFFORD. « The COPS Security Checker System ». Dans *Proceedings of the 14th National Computer Security Conference*, pages 372–385, October 1991.

[FOX 90] Kevin L. FOX, Ronda R. HENNING, Jonathan H. REED, et Richard P. SIMONIAN. « A Neural Network Approach Towards Intrusion Detection ». Rapport Technique, Harris Corporation, 1990.

[GAR 91] T.D. GARVEY et T.F. LUNT. « Model-based Intrusion Detection ». Dans *Proceedings of the 14th National Computer Security Conference*, October 1991.

[GAU 91] G. GAUDIN. « Gestion de sécurité en environnement hétérogène ». Dans *De nouvelles architectures pour les communications : les réseaux informatiques, qualité de service, sécurité et performances*, pages 73–80, octobre 1991.

[GEN 92] C.R GENT et C.P SHEPPARD. « Predicting Time Series by a Fully Connected Neural Network Trained by Back Propagation ». *Computing and Control Engineering Journal*, May 1992.

[HAB 92] Naji HABRA, Baudouin Le CHARLIER, Abdelaziz MOUNJI, et Isabelle MATHIEU. « ASAX : Software Architecture and Rule- Based Language for Universal Audit Trail Analysis ». Dans *European Symposium on Research in Computer Security (ESORICS)*, pages 435–450, 1992.

[HOL 75] John H. HOLLAND. « *Adaptation in Natural and Artificial Systems* ». Thèse de doctorat, University of Michigan, Ann Arbor, 1975.

[JAC 91] Kathleen A. JACKSON, David H. DUBOIS, et Cathy A. STALLING. « An Expert System Application for Network Intrusion Detection ». Dans *Proceedings of the 14th National Computer Security Conference*, 1991.

[JAV 93] H.S. JAVITZ, A. VALDES, T.F. LUNT, A. TAMARU, M. TYSON, et J. LOWRANCE. « Next Generation Intrusion Detection Expert System (NIDES) ». Rapport Technique A016-Rationales, SRI, 1993.

[JAV 94] Harold S. JAVITZ et Alfonso VALDES. « The NIDES Statistical Component Description and Justification ». Rapport Technique A010, SRI, 1994.

[KIM 91] J. KIMMINS. « Developing a Network Security Architecture : Concepts and Issues ». Dans *Proceedings of the SECURICOM'91 Conference*, march 1991.

[KUM 95] Sandeep KUMAR. « *Classification and Detection of Computer Intrusions* ». Thèse de doctorat, Purdue University, August 1995.

[LUN 88] T.F. LUNT et R. JAGANNATHAN. « A Prototype Real-Time Intrusion-Detection Expert System ». Dans *Proceedings of the IEEE Symposium on Security and Privacy*, pages 59–66, 1988.

[LUN 90a] Teresa LUNT. « IDES: An intelligent System for Detecting Intruders ». Dans *Computer Security, Threats and Countermeasures*, November 1990.

[LUN 90b] T.F. LUNT, A. TAMARU, F. GILHAM, R. JAGANNATHAN, C. JALALI, H.S. JAVITZ, A. VALDES, et P.G. NEUMANN. « A real-Time Intrusion-Detection Expert System ». Rapport Technique, SRI International, June 1990.

[LUN 90c] T.F. LUNT, A. TAMARU, F. GILHAM, R. JAGANNATHAN, P.G. NEUMANN, et C. JALALI. « IDES: A Progress Report ». Dans *Proceedings of the Computer Security Application Conference*, pages 273–285, 1990.

[Me92] Ludovic MÉ. « Audit de sécurité ». Rapport Technique 92-002, SUPÉLEC/LM, 1992.

[Me94] Ludovic MÉ. « *Audit de sécurité par algorithmes génétiques* ». Thèse de doctorat, Université de Rennes 1 - Numéro d'ordre 1069, 1994.

[MCA 90] N. MCAULIFFE, D. WOLCOTT, L. SCHAEFER, N. KELEM, B. HUBBARD, et T. HALUEY. « Is your Computer Being Misused? A Survey of Current Intrusion Detection System Technology ». Dans *Proceedings of the IEEE Computer Security Applications Conference*, pages 260–272, 1990.

[MUF 92] Alec D.E. MUFFETT. « Crack: A Sensible Password Checker for Unix ». from ftp:corton.inria.fr/CERT/tools/crack/crack4.1.tar, March 1992.

[POR 92] Phillip Andrew PORRAS. « A State Transition Analysis Tool For Intrusion Detection ». Master's thesis, University of California, Santa Barbara, 1992.

[ROL 94] Pierre ROLIN, Ludovic MÉ, et José VAZQUEZ. « Sécurité des systèmes informatiques ». *Réseaux et Informatique Repartie*, pages 31–74, 1994.

[SAF 93] David R. SAFFORD, Douglas Lee SCHALES, et David K. HESS. « The TAMU Security Package: an Ongoing response to the Internet Intruders in an Academic Environment ». Dans *Proceedings of the Fourth USENIX Security Symposium*, 1993.

[SEB 88] M.M. SEBRING, E. SHELLHOUSE, M.E. HANNA, et R.A. WHITEHURST. « Expert System in Intrusion Detection: A Case Study ». Dans *Proceedings of the 11th National Computer Security Conference*, pages 74–81, 1988.

[SMA 88] S.E. SMAHA. « Haystack: An Intrusion Detection System ». Dans *Proceedings of the 4th Aerospace Computer Security Application Conference*, pages 37–44, December 1988.

[STA 91] Timothy STARKWEATHER, S. MCDANIEL, K. MATHIAS, et C. Whitley DARRELL WHITLEY. « A comparison of Genetic Sequencing Operators ». Dans *Proceedings of the Fourth International Conference on Genetic Algorithms and their Applications*, pages 69–76, 1991.

[TSU 88] Gene TSUDIK. « An Expert System for Security Auditing ». Rapport Technique, IBM Los Angeles Scientific Software, 1988.

[TSU 90] Gene TSUDIK et R. SUMMERS. « AudES: An Expert System for Security Auditing ». Dans *Proceedings of the AAAI Conference on Innovative Application in Artificial Intelligence*, May 1990.

[VAC 89] H.S. VACCARO et G.E. LIEPINS. « Detection of Anomalous Computer Session Activity ». Dans *Proceedings of the IEEE Symposium on Security and Privacy*, May 1989.

[WIN 90] J.R. WINKLER. « A UNIX Prototype for Intrusion Detection and Anomaly Detection in Secure Networks ». Dans *Proceedings of the 13th National Computer Security Conference*, 1990.

[WIN 92] J.R. WINKLER. « Intrusion and Anomaly Detection; ISOA Update ». Dans *Proceedings of the 15th National Computer Security Conference*, 1992.